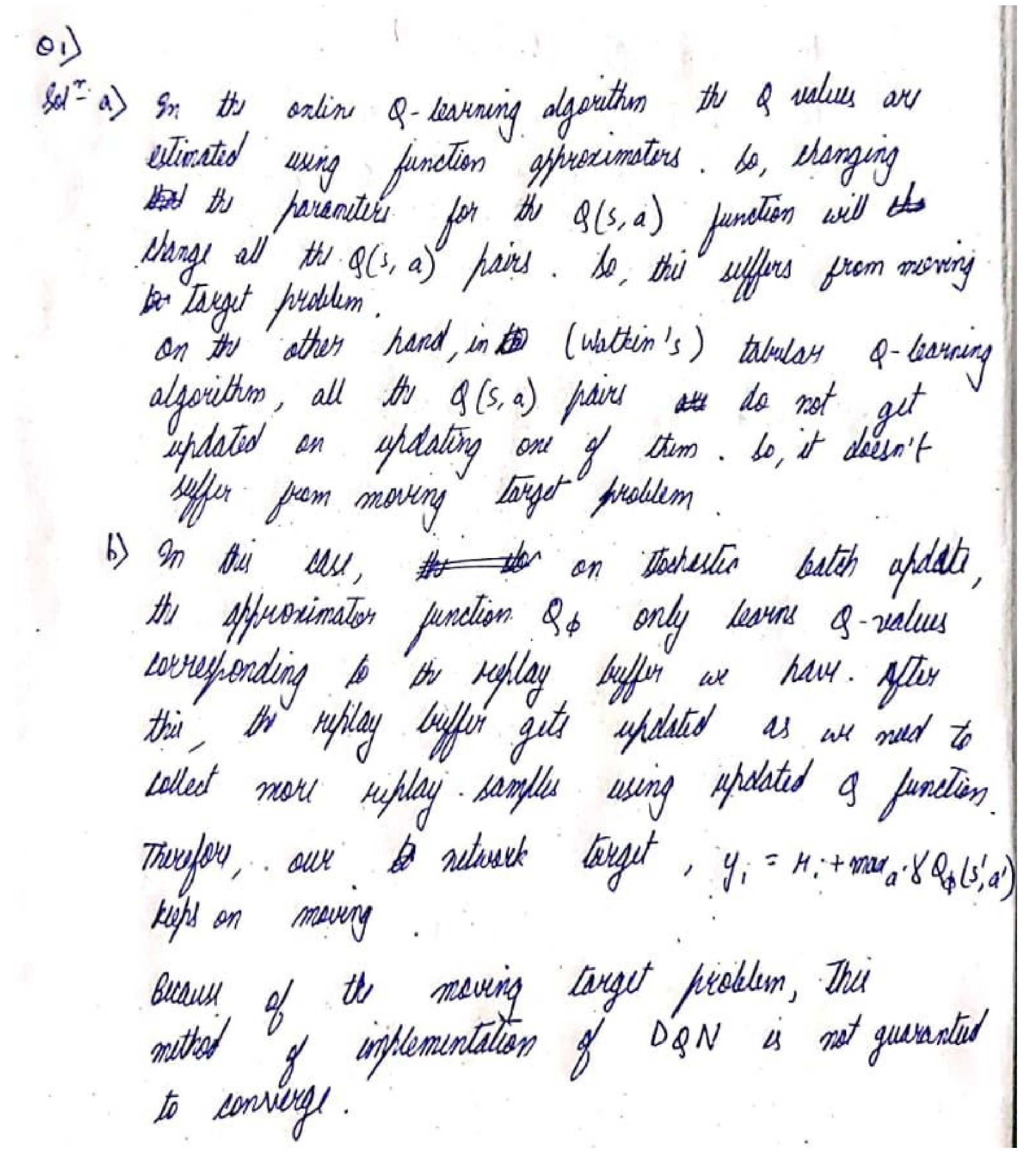
CS5500
Reinforcement Learning
Assignment 3

Name - Raktim Gautam Goswami
Roll Number - EE17BTECH11051

Note:- Since, I am doing a project for CS5500, so, according to the information given, I have attempted only Problem 1 (5 points) and Problem 3 (25 points), giving a total of 30 points.

Problem 1
Solution -



Q1)

Solⁿ:- a) In the online Q-learning algorithm the Q values are estimated using function approximators. So, changing ~~the~~ the parameters for the Q(s, a) function will ~~the~~ change all the Q(s, a) pairs. So, this suffers from moving ~~to~~ target problem.

On the other hand, in ~~the~~ (Watkin's) tabular Q-learning algorithm, all the Q(s, a) pairs ~~are~~ do not get updated on updating one of them. So, it doesn't suffer from moving target problem.

b) In this case, ~~the the~~ on stochastic batch update, the approximator function Q_ϕ only learns Q-values corresponding to the replay buffer we have. After this, the replay buffer gets updated as we need to collect more replay samples using updated Q function.

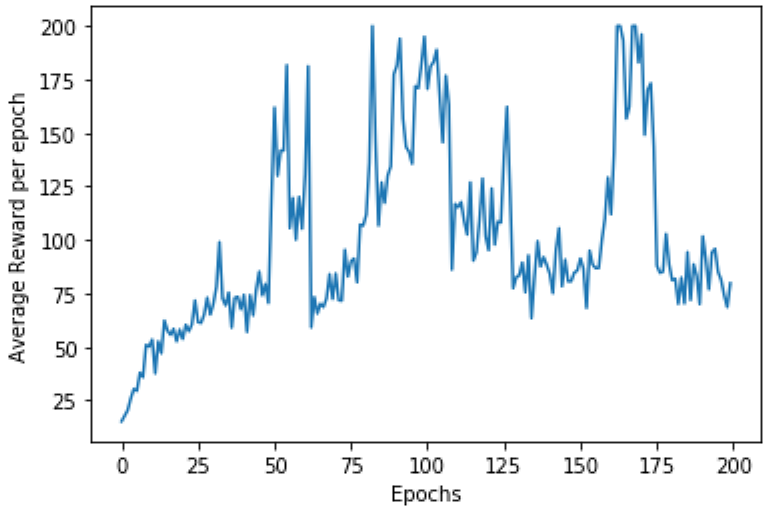
Therefore, our ~~Q~~ network target, $y_i = r_i + \max_{a'} \gamma Q_\phi(s', a')$ keeps on moving.

Because of the moving target problem, this method of implementation of DQN is not guaranteed to converge.

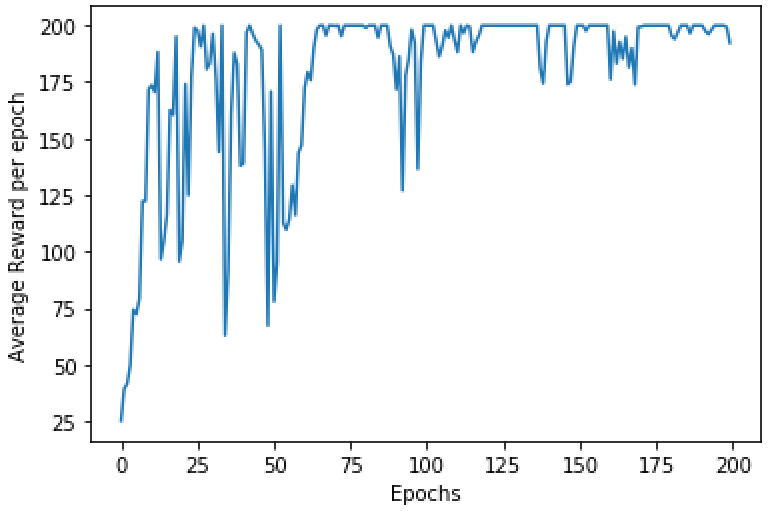
Problem 3
Solution-

The codes for the question have also been submitted. Instructions on running the code are in the ReadME.txt file.

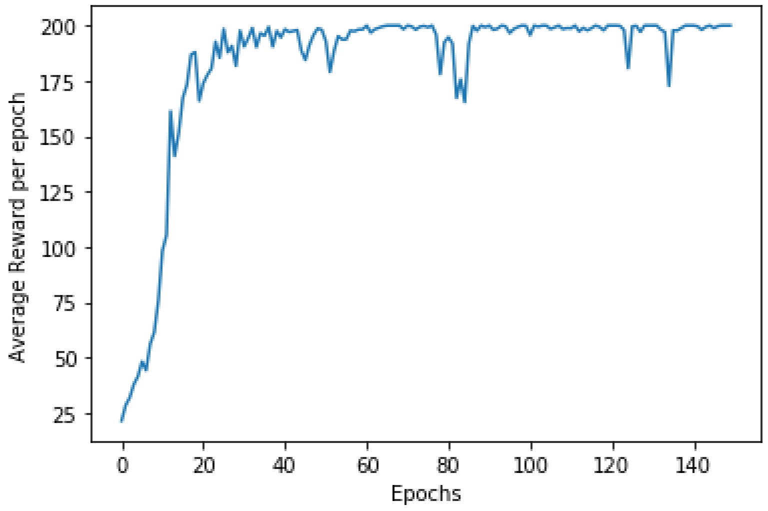
a) The reward function of the respective environments are as follows-
    i) Cart Pole - The agent gets a reward of +1 for every time step it stays before ending the episode
    ii) Lunar Lander - Reward for moving from the top of the screen to landing pad and zero speed is about 100..140 points. As the lander moves away from the landing pad, it starts losing reward. If the lander comes to rest at the desired position, it gets an addition +100 points. If it crashes, it gets -100 points additional. Each leg contact with ground gives the lander additional +10 points. Firing the main engine gives -0.3 points and firing the side engines gives -0.03 points additional. If the problem is solved, the agent gets an additional 200 points.
b) Comparison of learning curves are as follows -
    i) Cart pole
        1) Without Reward to go and without Advantage normalisation



2) With Reward to go and without Advantage normalisation

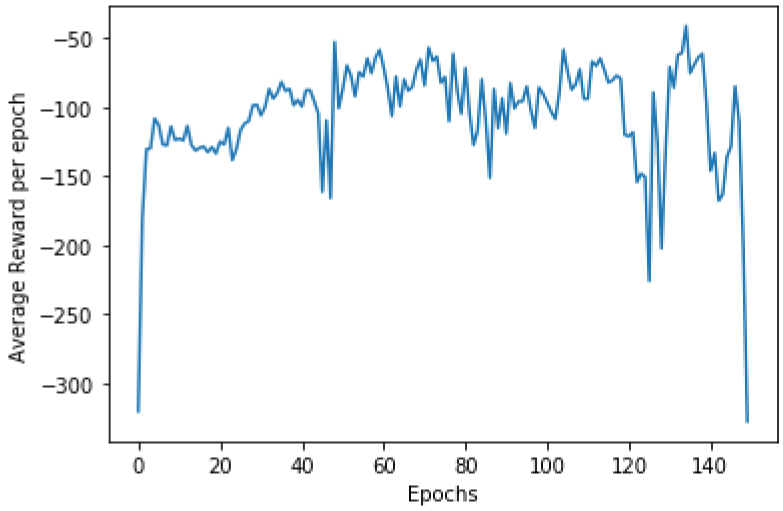


3) With Reward to go and with Advantage normalisation

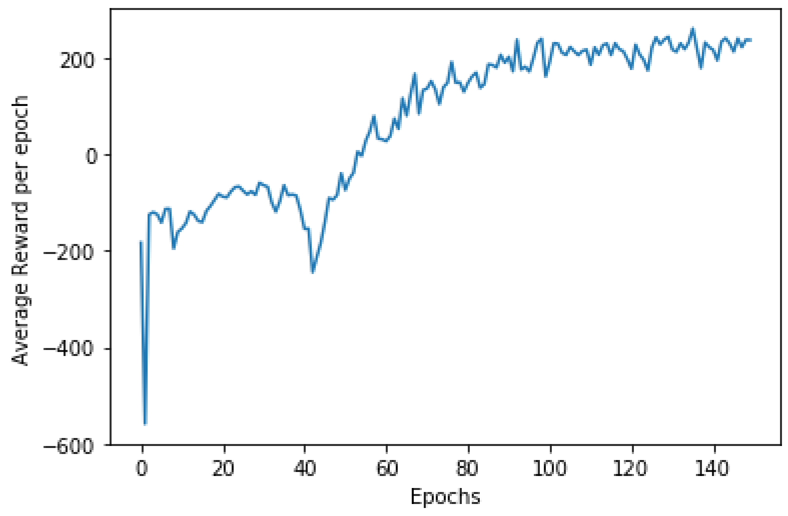


ii) Lunar Lander

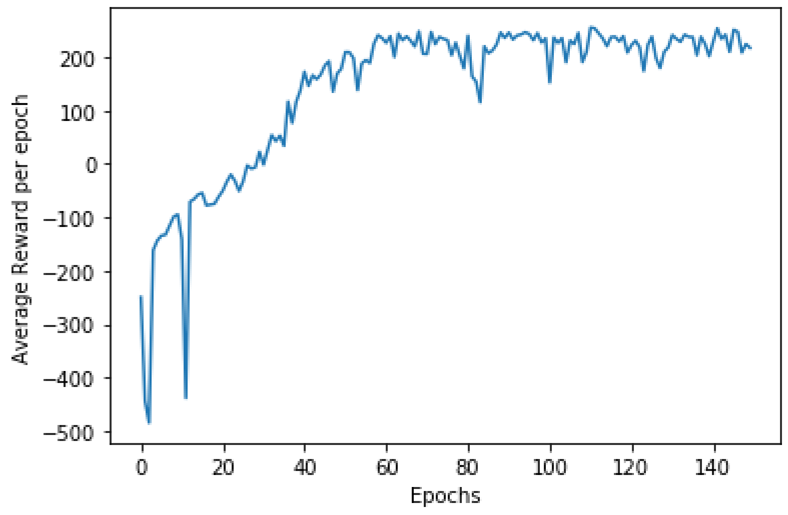
1) Without Reward to go and without Advantage normalisation



2) With Reward to go and without Advantage normalisation



3) With Reward to go and with Advantage normalisation



It can be clearly seen in both the environments that using Reward To Go gave better performance than without using it. This is because using Reward To Go reduces the variance of the gradient.
Using Advantage Normalisation with Reward To Go gave the best result as in this case, the bias of the gradient is also reduced.

c) To report the batch size, the code was run many times using different batch sizes. One of the clear indications of the effect of batch size is that on increasing batch size, the algorithm took much less number of epochs to converge. However, the total time taken was approximately the same in both cases. Using a very small number in batch size means that the loss contour changes very frequently. Because of this, the optimiser might not be able to find a global minimum. On the other hand, using a very high number in batch size means that the loss contour will not change much over batches. So, the optimiser might be stuck in some local minima. So, we need to choose a number which is not very high and at the same time, not very low.

Q3)

Sol^n d) $\nabla_\theta J(\theta) = E_{\pi_\theta}\left[\sum_{t=0}^{\infty} \nabla_\theta \log \pi_\theta(a_t|s_t)\, \Psi_t\right]$, where $\Psi_t = G_{t:\infty} - b(s_t)$

$$\nabla_\theta J(\theta) = E_{\pi_\theta}\left[\sum_{t=0}^{\infty} \nabla_\theta \log \pi_\theta(a_t|s_t)\,(G_{t:\infty} - b(s_t))\right]$$

$$= \underset{\tau\sim\pi_\theta(\tau)}{E}\left[\nabla_\theta \log \pi_\theta(\tau)\,(G_{t:\infty}(\tau) - b)\right]$$

Now,

$$Var = \underset{\tau\sim\pi_\theta(\tau)}{E}\left[(\nabla_\theta \log\pi_\theta(\tau)\,(G_{t:\infty}(\tau) - b))^2\right] - \underset{\tau\sim\pi_\theta(\tau)}{E}\left[\nabla_\theta \log \pi_\theta(\tau)\, G_{t:\infty}\right]$$

For minimum variance,

$$\frac{\partial Var}{\partial b} = 0$$

$$\Rightarrow -2\underset{\tau\sim\pi_\theta(\tau)}{E}\left[\nabla_\theta \log^2 \pi_\theta(\tau)\, G_{t:\infty}(\tau)\right] + 2b\underset{\tau\sim\pi_\theta(\tau)}{E}\left[\nabla_\theta \log^2 \pi_\theta(\tau)\right] = 0$$

$$\Rightarrow b = \frac{\underset{\tau\sim\pi_\theta(\tau)}{E}\left[\nabla_\theta \log^2 \pi_\theta(\tau)\, G_{t:\infty}(\tau)\right]}{\underset{\tau\sim\pi_\theta(\tau)}{E}\left[\nabla_\theta \log \pi_\theta^2(\tau)\right]}$$

$$= \frac{E_{\pi_\theta}\left[\nabla_\theta \log \pi(a_t|s_t)^2\, G_{t:\infty}(\tau)\right]}{E_{\pi_\theta}\left[\nabla_\theta \log \pi(a_t|s_t)^2\right]}$$

d)